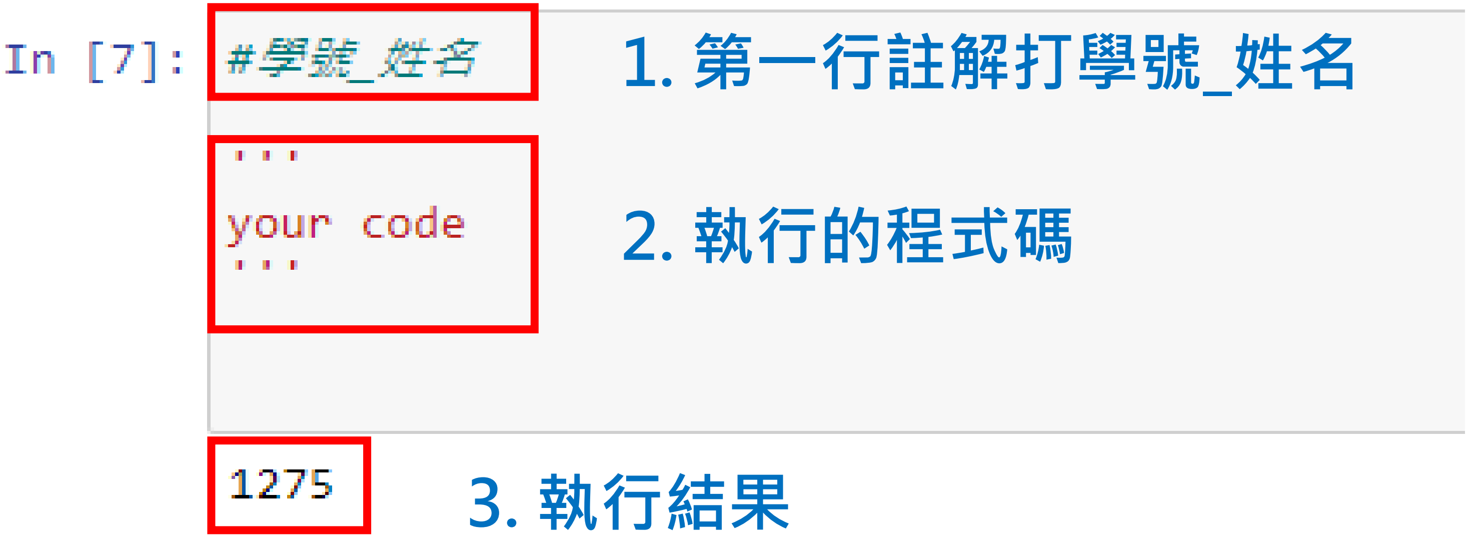
# HW2

# 繳交方式

- Capture the result, including your student number, name, code, output in the word document.
- 繳交格式:pdf或word檔
- 檔名:s學號_hw2



```
In [7]: #學號_姓名
        '''
        your code
        '''

        1275
```

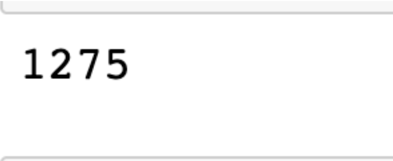
1. 第一行註解打學號_姓名

2. 執行的程式碼

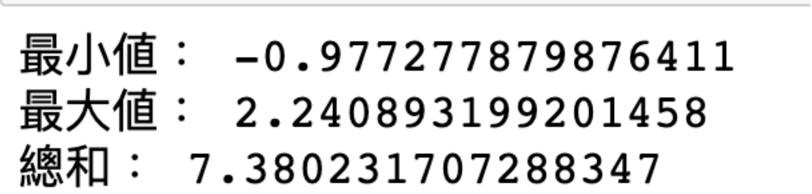
3. 執行結果

# Numpy（共四題)

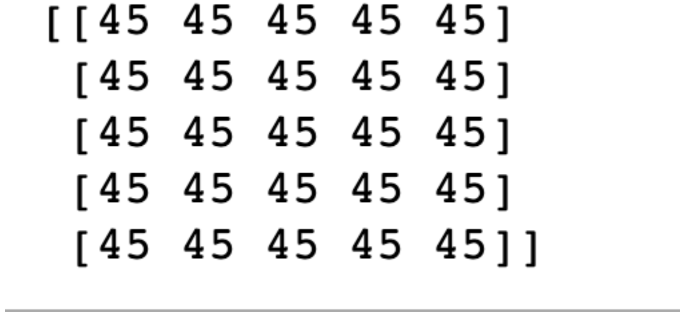
- Q1. 使用np.array來製作1到50的陣列，計算從1到50的自然數總和，並顯示最後的計算成果
- Q2. 亂數種子為0，依標準常態生成10個亂數，並製作陣列，求得其中的最小值、最大值、總和
- Q3. 製作所有元素均為3的5列5行矩陣，計算該矩陣的平方



```
1275
```



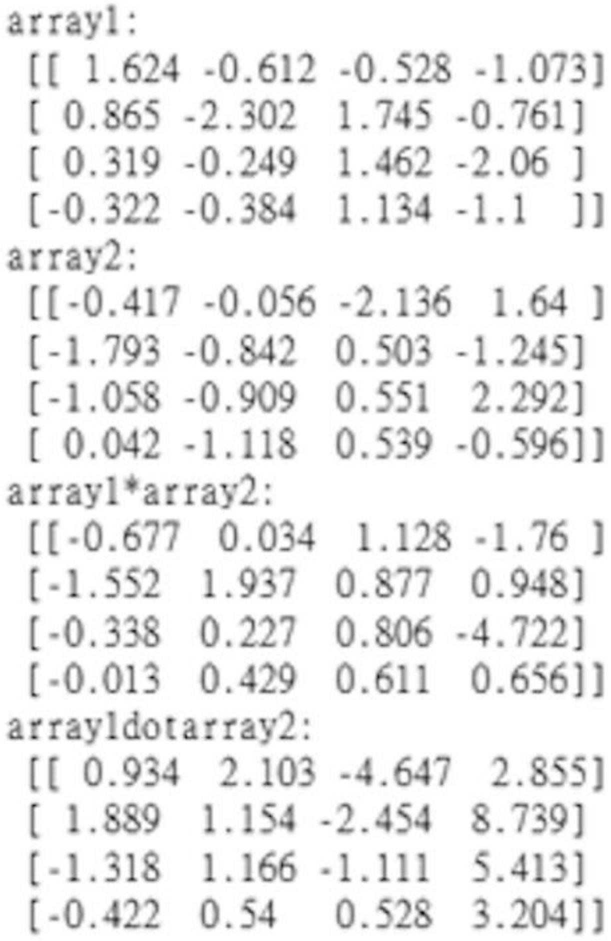
```
最小值：  -0.977277879876411
最大值：  2.240893199201458
總和：  7.380231707288347
```



```
[[45 45 45 45 45]
 [45 45 45 45 45]
 [45 45 45 45 45]
 [45 45 45 45 45]
 [45 45 45 45 45]]
```

# Numpy

- Q4.
  - 使用標準常態、種子為1，產生一個4*4的矩陣
  - 使用標準常態、種子為2，產生一個4*4的矩陣
  - 計算矩陣元素相乘
  - 計算矩陣相乘



```
array1:
 [[ 1.624 -0.612 -0.528 -1.073]
 [ 0.865 -2.302  1.745 -0.761]
 [ 0.319 -0.249  1.462 -2.06 ]
 [-0.322 -0.384  1.134 -1.1  ]]
array2:
 [[-0.417 -0.056 -2.136  1.64 ]
 [-1.793 -0.842  0.503 -1.245]
 [-1.058 -0.909  0.551  2.292]
 [ 0.042 -1.118  0.539 -0.596]]
array1*array2:
 [[-0.677  0.034  1.128 -1.76 ]
 [-1.552  1.937  0.877  0.948]
 [-0.338  0.227  0.806 -4.722]
 [-0.013  0.429  0.611  0.656]]
array1dotarray2:
 [[ 0.934  2.103 -4.647  2.855]
 [ 1.889  1.154 -2.454  8.739]
 [-1.318  1.166 -1.111  5.413]
 [-0.422  0.54   0.528  3.204]]
```

]

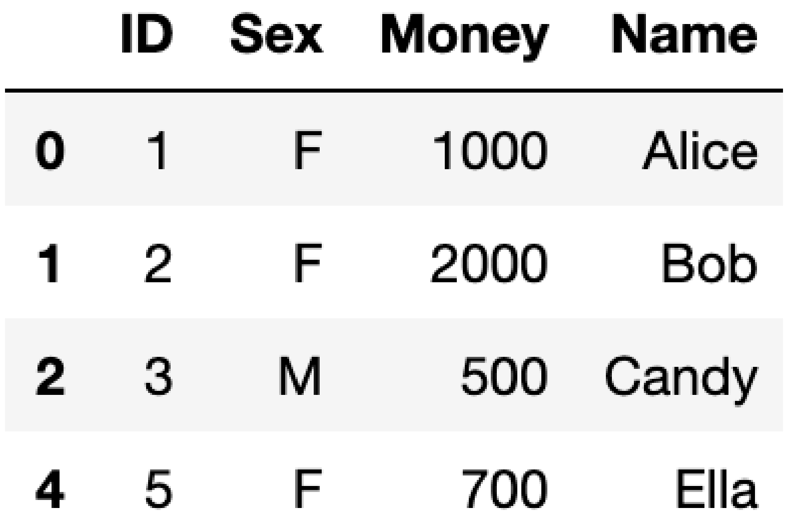
# Pandas

- 輸入以下的資料。

```
from pandas import Series,DataFrame
import pandas as pd

attri_data1 = {'ID':['1','2','3','4','5'],
            'Sex':['F','F','M','M','F'],
            'Money':[1000,2000,500,300,700],
            'Name':['Alice','Bob','Candy','David','Ella']}

attri_data_frame1 = DataFrame
```

# Pandas (共四題)

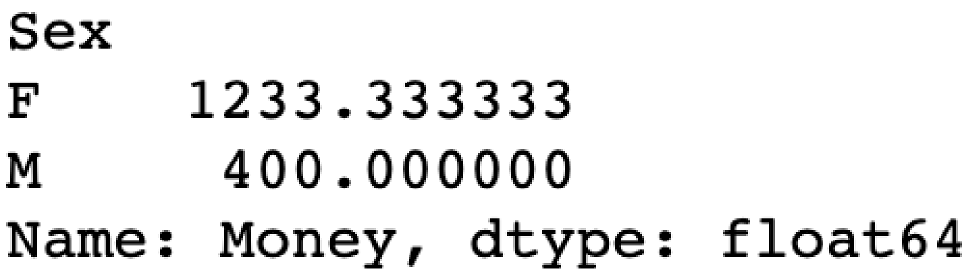
- Q1: 請找出Money為500以上的人(包含500)，並顯示。



|   | ID | Sex | Money | Name |
|---|---|---|---|---|
| **0** | 1 | F | 1000 | Alice |
| **1** | 2 | F | 2000 | Bob |
| **2** | 3 | M | 500 | Candy |
| **4** | 5 | F | 700 | Ella |

# Pandas

- Q2: 請計算男女的平均Money。



```
Sex
F    1233.333333
M     400.000000
Name: Money, dtype: float64
```

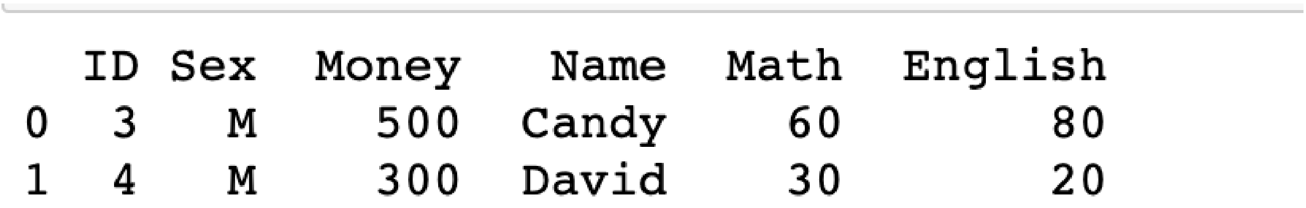
# Pandas

- 輸入attri_data2的資料。

```
attri_data2 = {'ID':['3','4','7'],
             'Math':[60,30,40],
             'English':[80,20,30]}

attri_data_frame2 = DataFrame(attri_data2)
```

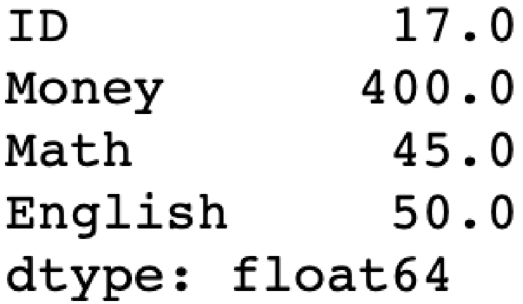
# Pandas

- Q3: 將 attri_data1 與 attri_data2 合併，並顯示。



```
   ID Sex  Money   Name  Math  English
0   3   M    500  Candy    60       80
1   4   M    300  David    30       20
```

- Q4: 並將Money、Math、English進行平均，並顯示。



```
ID           17.0
Money       400.0
Math         45.0
English      50.0
dtype: float64
```

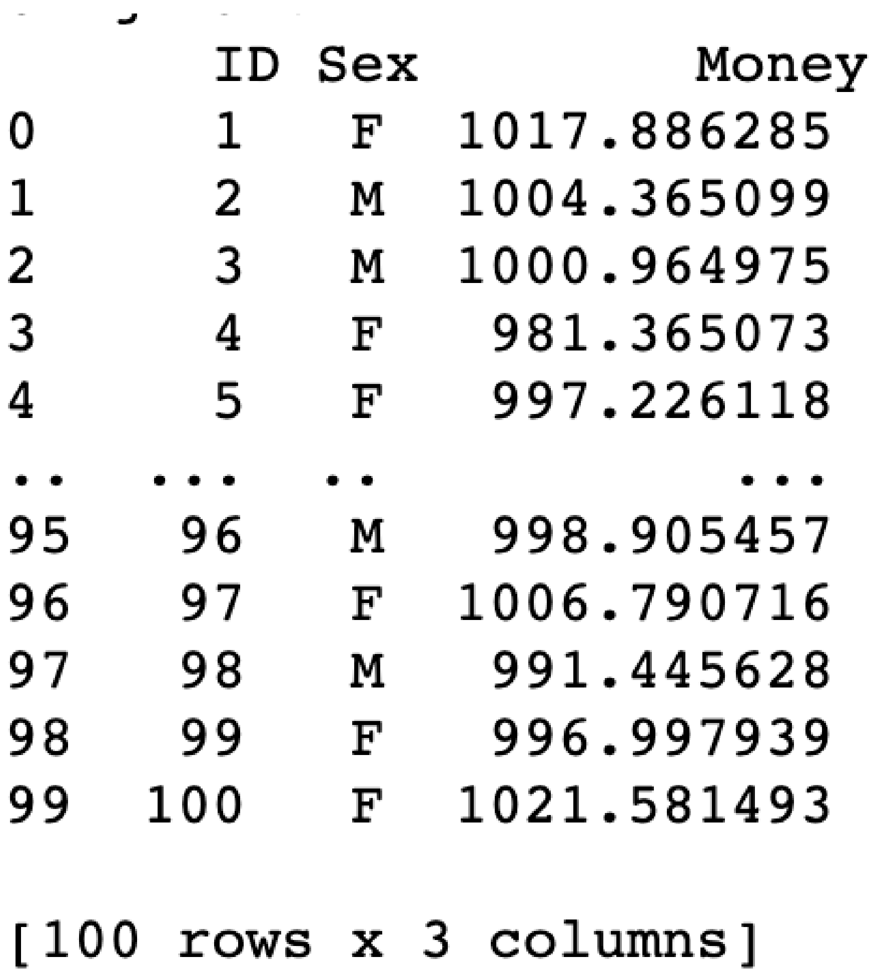
# Pandas Advanced

- 在100筆資料中，亂數產生性別
- np.random.seed(2)
- array2 = np.random.randint(2, size=100)
- 0:Female
- 1:Male

- 亂數產生錢
- np.random.seed(3)
- array3=np.random.normal(1000, 10, size=100)

# Pandas Advanced: 共三題

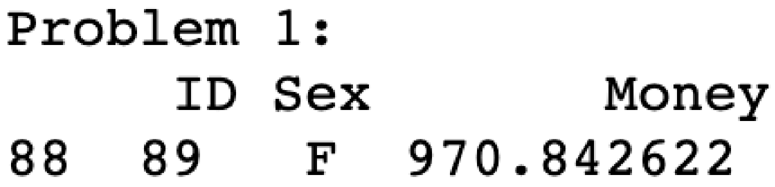
產生的矩陣



```
           ID Sex        Money
0       1   F  1017.886285
1       2   M  1004.365099
2       3   M  1000.964975
3       4   F   981.365073
4       5   F   997.226118
..    ...  ..          ...
95     96   M   998.905457
96     97   F  1006.790716
97     98   M   991.445628
98     99   F   996.997939
99    100   F  1021.581493

[100 rows x 3 columns]
```

# Q1

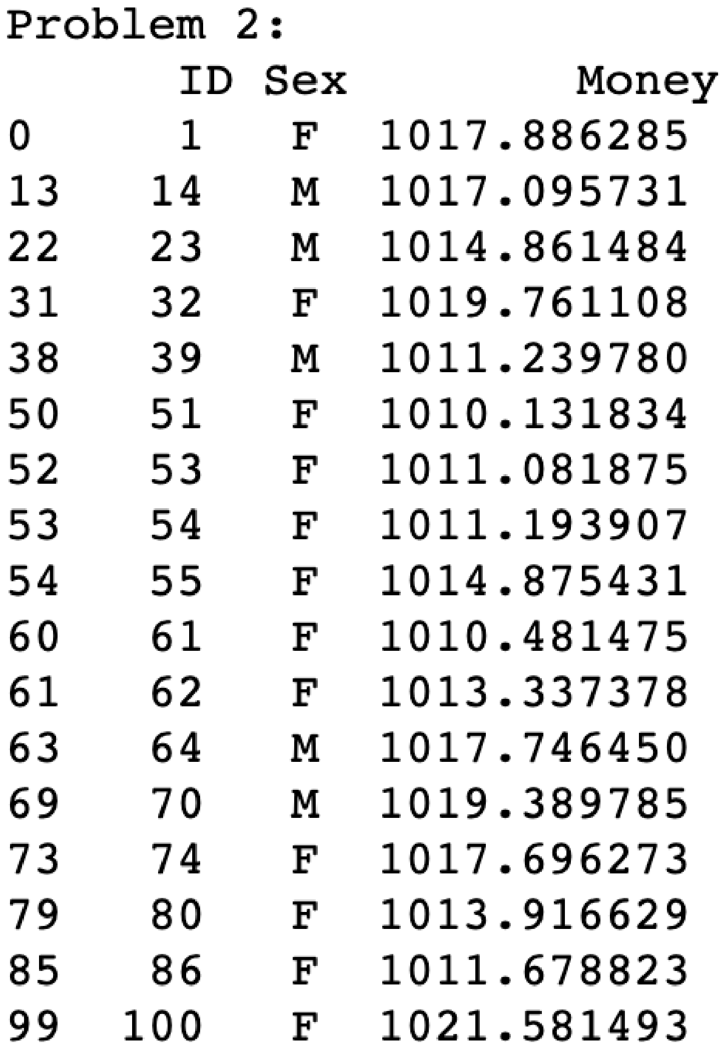
- 取錢的最小值



```
Problem 1:
     ID Sex         Money
88   89   F  970.842622
```

# Q2

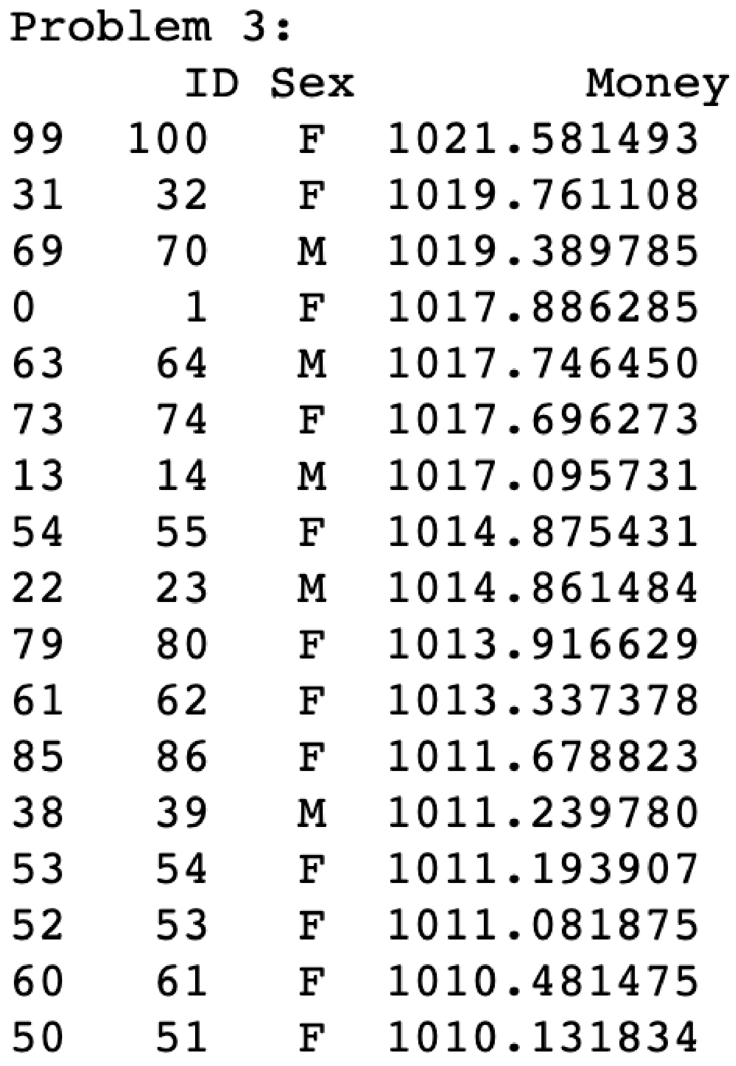
- 取錢大於1010



```
Problem 2:
         ID Sex         Money
0         1   F   1017.886285
13       14   M   1017.095731
22       23   M   1014.861484
31       32   F   1019.761108
38       39   M   1011.239780
50       51   F   1010.131834
52       53   F   1011.081875
53       54   F   1011.193907
54       55   F   1014.875431
60       61   F   1010.481475
61       62   F   1013.337378
63       64   M   1017.746450
69       70   M   1019.389785
73       74   F   1017.696273
79       80   F   1013.916629
85       86   F   1011.678823
99      100   F   1021.581493
```

# Q3

- 第二題的結果用”Money“，由大到小排序

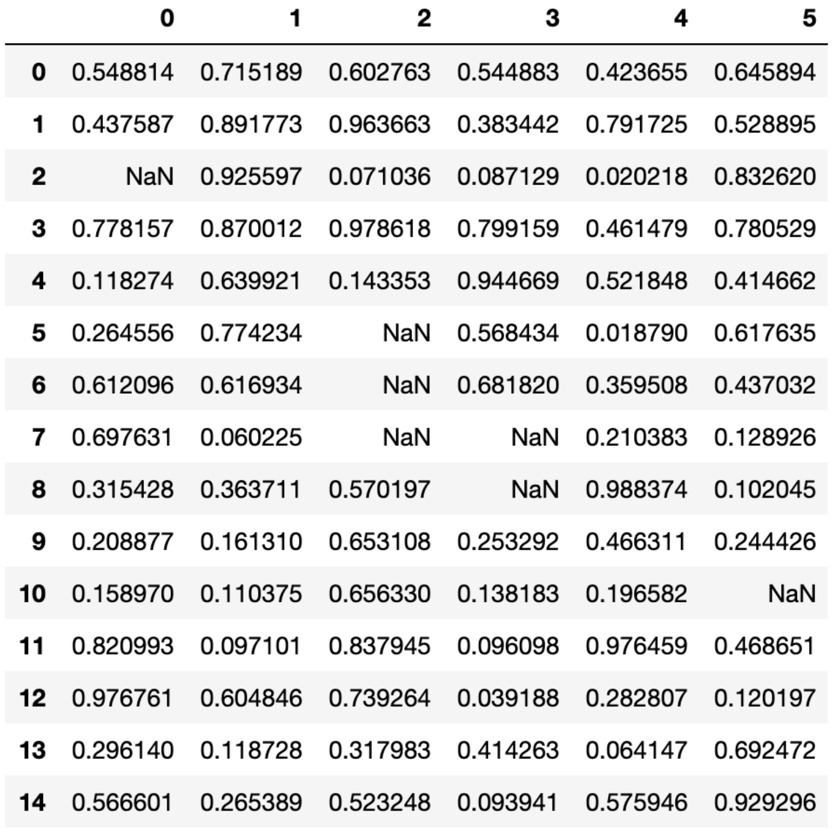


```
Problem 3:
        ID Sex          Money
99   100   F   1021.581493
31    32   F   1019.761108
69    70   M   1019.389785
0      1   F   1017.886285
63    64   M   1017.746450
73    74   F   1017.696273
13    14   M   1017.095731
54    55   F   1014.875431
22    23   M   1014.861484
79    80   F   1013.916629
61    62   F   1013.337378
85    86   F   1011.678823
38    39   M   1011.239780
53    54   F   1011.193907
52    53   F   1011.081875
60    61   F   1010.481475
50    51   F   1010.131834
```

# Fillna (共三題)

- 假設值為NaN(NA)的部分即為遺漏資料

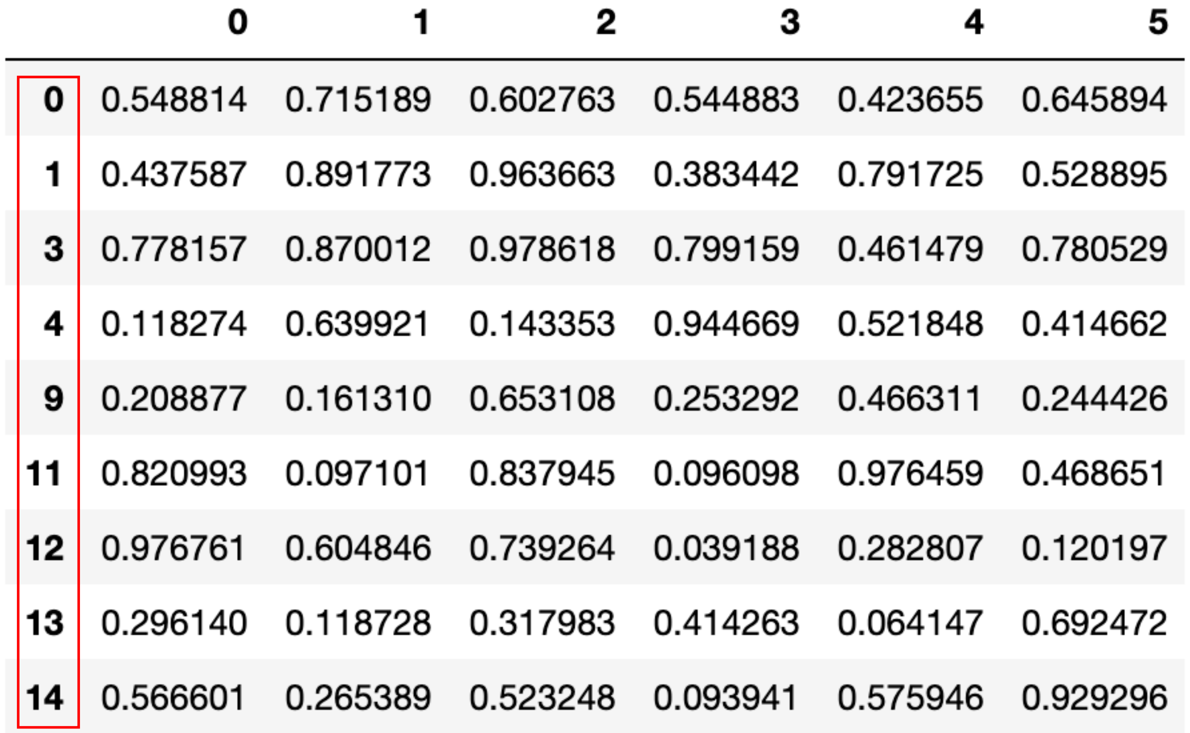
```
import numpy as np
from numpy import nan as NA
import pandas as pd
import numpy.random as random

random.seed(0)
df2 = pd.DataFrame(np.random.rand(15, 6))

df2.iloc[2,0] = NA
df2.iloc[5:8,2] = NA
df2.iloc[7:9,3] = NA
df2.iloc[10,5] = NA

df2
```



| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| **0** | 0.548814 | 0.715189 | 0.602763 | 0.544883 | 0.423655 | 0.645894 |
| **1** | 0.437587 | 0.891773 | 0.963663 | 0.383442 | 0.791725 | 0.528895 |
| **2** | NaN | 0.925597 | 0.071036 | 0.087129 | 0.020218 | 0.832620 |
| **3** | 0.778157 | 0.870012 | 0.978618 | 0.799159 | 0.461479 | 0.780529 |
| **4** | 0.118274 | 0.639921 | 0.143353 | 0.944669 | 0.521848 | 0.414662 |
| **5** | 0.264556 | 0.774234 | NaN | 0.568434 | 0.018790 | 0.617635 |
| **6** | 0.612096 | 0.616934 | NaN | 0.681820 | 0.359508 | 0.437032 |
| **7** | 0.697631 | 0.060225 | NaN | NaN | 0.210383 | 0.128926 |
| **8** | 0.315428 | 0.363711 | 0.570197 | NaN | 0.988374 | 0.102045 |
| **9** | 0.208877 | 0.161310 | 0.653108 | 0.253292 | 0.466311 | 0.244426 |
| **10** | 0.158970 | 0.110375 | 0.656330 | 0.138183 | 0.196582 | NaN |
| **11** | 0.820993 | 0.097101 | 0.837945 | 0.096098 | 0.976459 | 0.468651 |
| **12** | 0.976761 | 0.604846 | 0.739264 | 0.039188 | 0.282807 | 0.120197 |
| **13** | 0.296140 | 0.118728 | 0.317983 | 0.414263 | 0.064147 | 0.692472 |
| **14** | 0.566601 | 0.265389 | 0.523248 | 0.093941 | 0.575946 | 0.929296 |

# Fillna

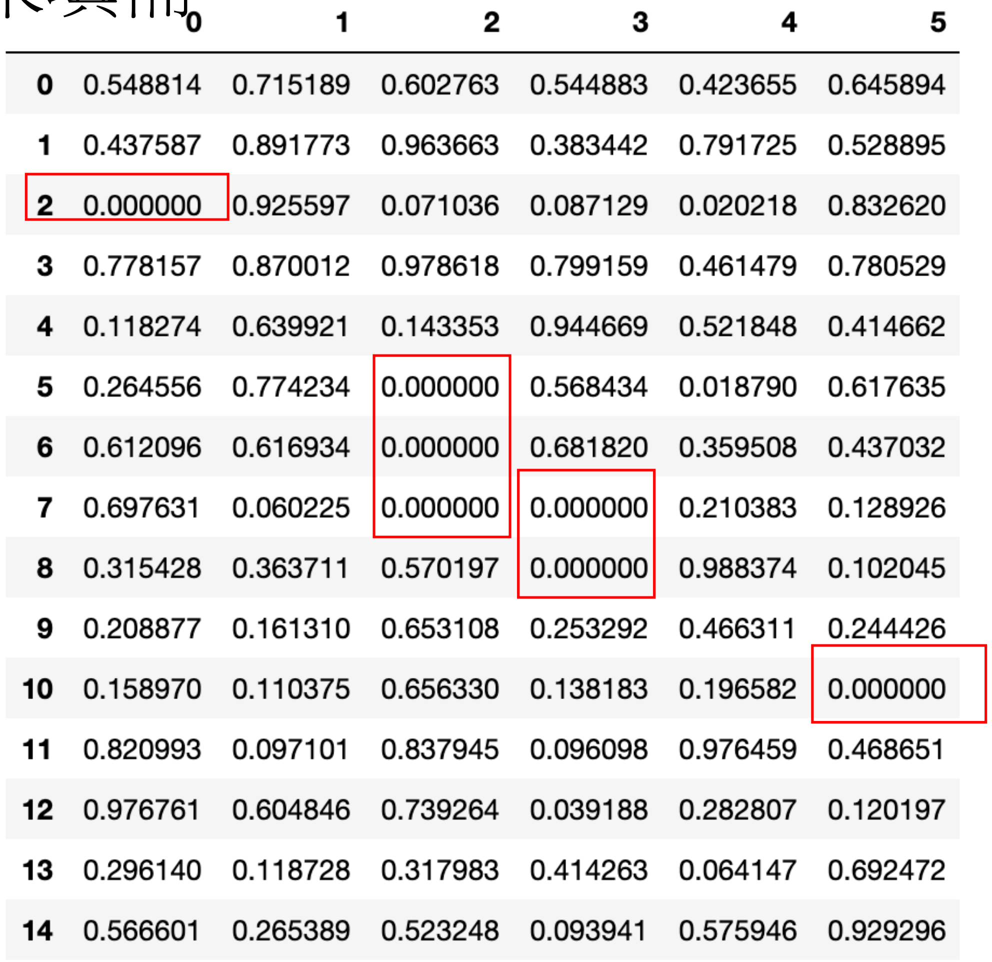
- Q1: 將有NaN的列刪除



| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| **0** | 0.548814 | 0.715189 | 0.602763 | 0.544883 | 0.423655 | 0.645894 |
| **1** | 0.437587 | 0.891773 | 0.963663 | 0.383442 | 0.791725 | 0.528895 |
| **3** | 0.778157 | 0.870012 | 0.978618 | 0.799159 | 0.461479 | 0.780529 |
| **4** | 0.118274 | 0.639921 | 0.143353 | 0.944669 | 0.521848 | 0.414662 |
| **9** | 0.208877 | 0.161310 | 0.653108 | 0.253292 | 0.466311 | 0.244426 |
| **11** | 0.820993 | 0.097101 | 0.837945 | 0.096098 | 0.976459 | 0.468651 |
| **12** | 0.976761 | 0.604846 | 0.739264 | 0.039188 | 0.282807 | 0.120197 |
| **13** | 0.296140 | 0.118728 | 0.317983 | 0.414263 | 0.064147 | 0.692472 |
| **14** | 0.566601 | 0.265389 | 0.523248 | 0.093941 | 0.575946 | 0.929296 |

# Fillna

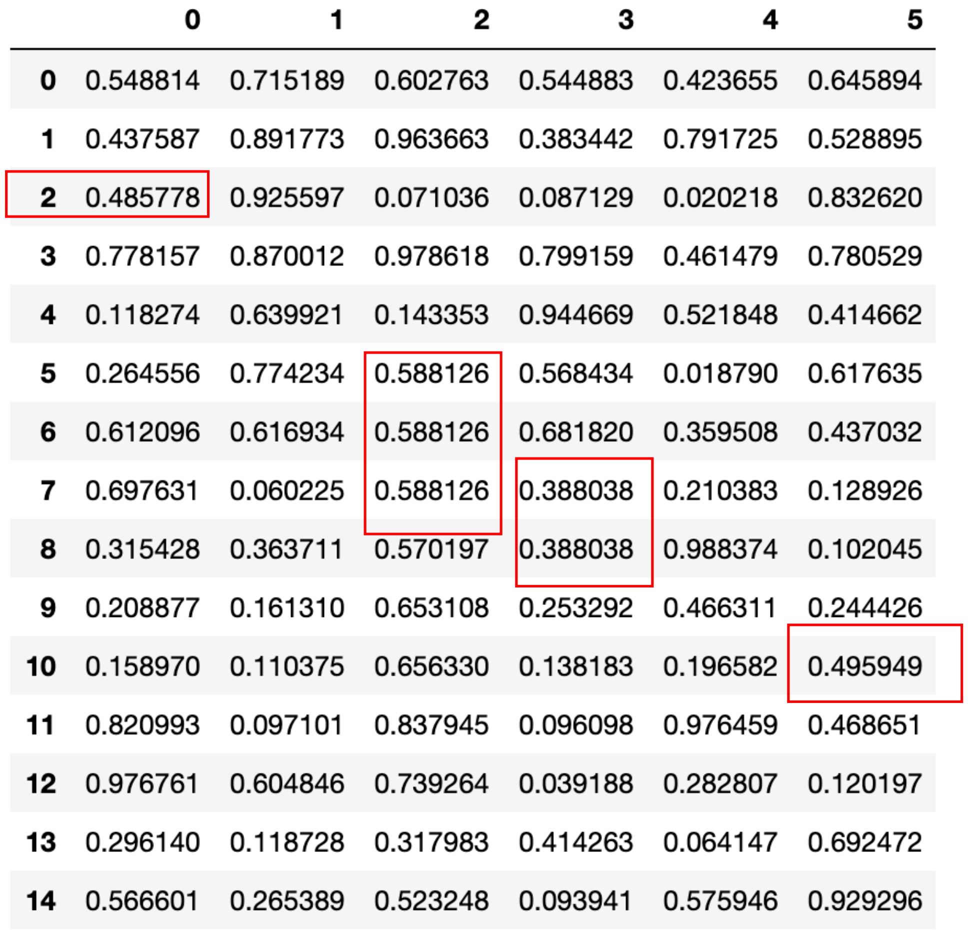
- Q2:請將NaN以0來填補



|  | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| **0** | 0.548814 | 0.715189 | 0.602763 | 0.544883 | 0.423655 | 0.645894 |
| **1** | 0.437587 | 0.891773 | 0.963663 | 0.383442 | 0.791725 | 0.528895 |
| **2** | 0.000000 | 0.925597 | 0.071036 | 0.087129 | 0.020218 | 0.832620 |
| **3** | 0.778157 | 0.870012 | 0.978618 | 0.799159 | 0.461479 | 0.780529 |
| **4** | 0.118274 | 0.639921 | 0.143353 | 0.944669 | 0.521848 | 0.414662 |
| **5** | 0.264556 | 0.774234 | 0.000000 | 0.568434 | 0.018790 | 0.617635 |
| **6** | 0.612096 | 0.616934 | 0.000000 | 0.681820 | 0.359508 | 0.437032 |
| **7** | 0.697631 | 0.060225 | 0.000000 | 0.000000 | 0.210383 | 0.128926 |
| **8** | 0.315428 | 0.363711 | 0.570197 | 0.000000 | 0.988374 | 0.102045 |
| **9** | 0.208877 | 0.161310 | 0.653108 | 0.253292 | 0.466311 | 0.244426 |
| **10** | 0.158970 | 0.110375 | 0.656330 | 0.138183 | 0.196582 | 0.000000 |
| **11** | 0.820993 | 0.097101 | 0.837945 | 0.096098 | 0.976459 | 0.468651 |
| **12** | 0.976761 | 0.604846 | 0.739264 | 0.039188 | 0.282807 | 0.120197 |
| **13** | 0.296140 | 0.118728 | 0.317983 | 0.414263 | 0.064147 | 0.692472 |
| **14** | 0.566601 | 0.265389 | 0.523248 | 0.093941 | 0.575946 | 0.929296 |

# Fillna

- Q3:將NaN以各行的平均值來填補



|  | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| **0** | 0.548814 | 0.715189 | 0.602763 | 0.544883 | 0.423655 | 0.645894 |
| **1** | 0.437587 | 0.891773 | 0.963663 | 0.383442 | 0.791725 | 0.528895 |
| **2** | 0.485778 | 0.925597 | 0.071036 | 0.087129 | 0.020218 | 0.832620 |
| **3** | 0.778157 | 0.870012 | 0.978618 | 0.799159 | 0.461479 | 0.780529 |
| **4** | 0.118274 | 0.639921 | 0.143353 | 0.944669 | 0.521848 | 0.414662 |
| **5** | 0.264556 | 0.774234 | 0.588126 | 0.568434 | 0.018790 | 0.617635 |
| **6** | 0.612096 | 0.616934 | 0.588126 | 0.681820 | 0.359508 | 0.437032 |
| **7** | 0.697631 | 0.060225 | 0.588126 | 0.388038 | 0.210383 | 0.128926 |
| **8** | 0.315428 | 0.363711 | 0.570197 | 0.388038 | 0.988374 | 0.102045 |
| **9** | 0.208877 | 0.161310 | 0.653108 | 0.253292 | 0.466311 | 0.244426 |
| **10** | 0.158970 | 0.110375 | 0.656330 | 0.138183 | 0.196582 | 0.495949 |
| **11** | 0.820993 | 0.097101 | 0.837945 | 0.096098 | 0.976459 | 0.468651 |
| **12** | 0.976761 | 0.604846 | 0.739264 | 0.039188 | 0.282807 | 0.120197 |
| **13** | 0.296140 | 0.118728 | 0.317983 | 0.414263 | 0.064147 | 0.692472 |
| **14** | 0.566601 | 0.265389 | 0.523248 | 0.093941 | 0.575946 | 0.929296 |